BSCCS/2024/31978 FAITH WANJA GICHURE

QUESTION 1

PLOTTING A LINE GRAPH OF TEMPERATURE READINGS OVER A WEEK: 20,22,19,23,21,24,20

#CODE

import matplotlib.pyplot as plt

temperatures = [20, 22, 19, 23, 21, 24, 20] # Temperature data

days = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'] # Days of the week

# Create the line graph

plt.plot(days, temperatures, marker='o', linestyle='-', color='blue')

plt.title('Temperature Readings Over a Week')

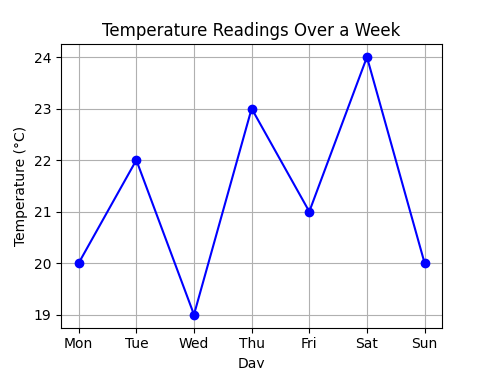
plt.xlabel('Day')

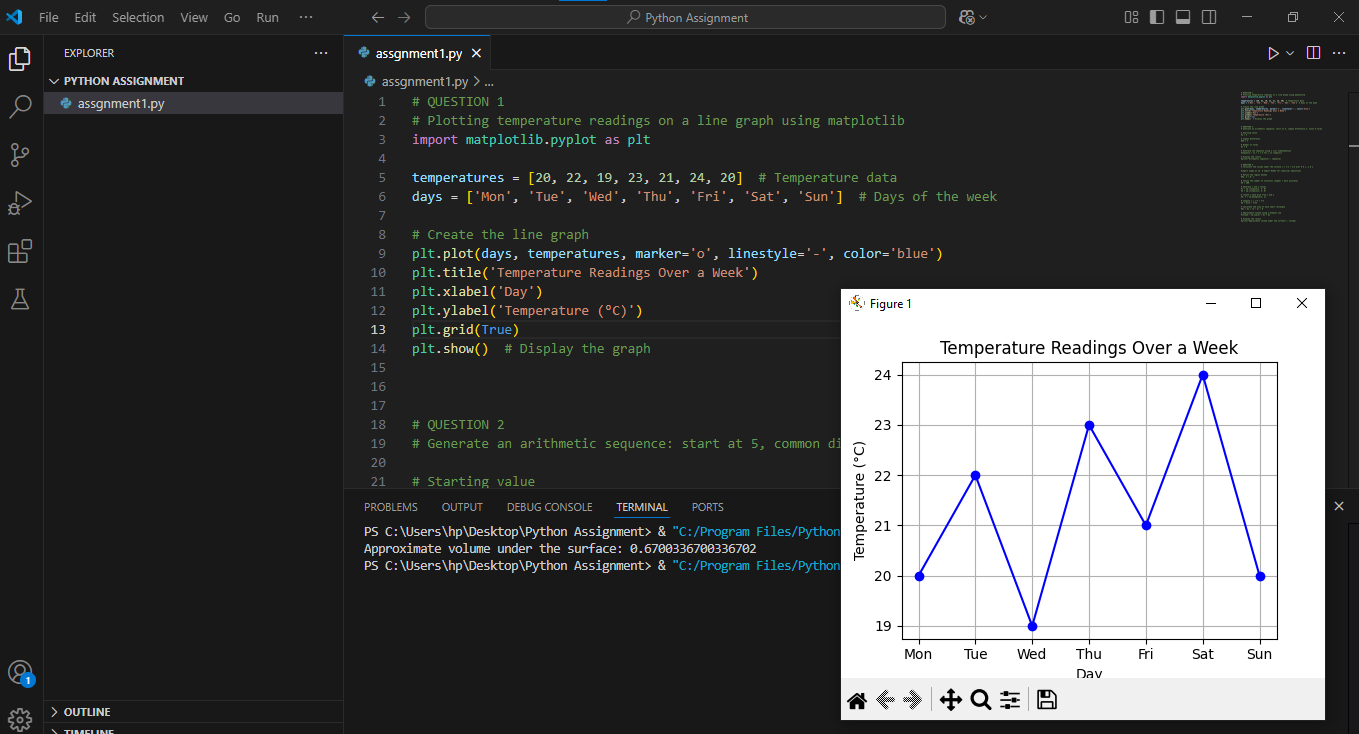
plt.ylabel('Temperature (°C)')

plt.grid(True)

plt.show() # Display the graph

Output





QUESTION 2

ARITHMETIC SEQUENCE GENERATOR STARTING AT 5 WITH A COMMON DIFFERENCE OF 3 FOR 8 TERMS

#CODE

# Starting value

a = 5

# Common difference

d = 3

# Number of terms

n = 8

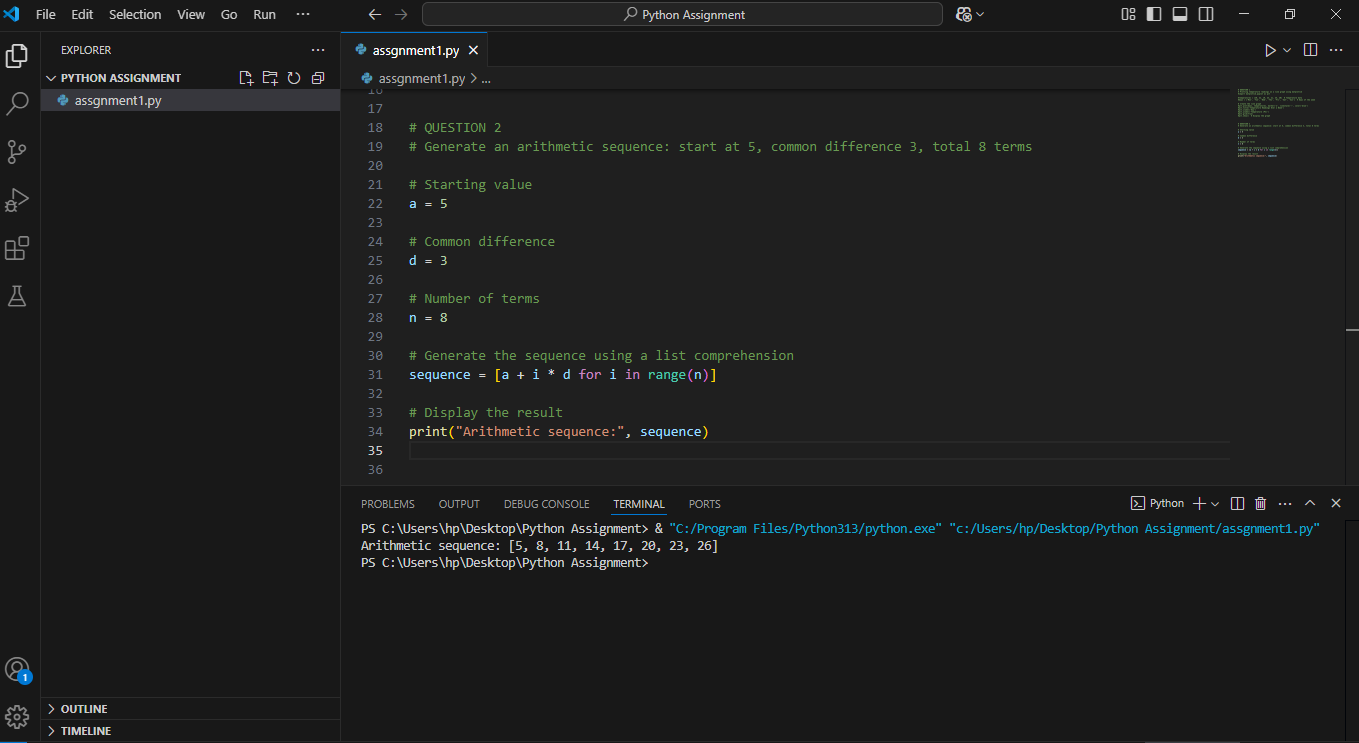
# Generate the sequence using a list comprehension

sequence = [a + i \* d for i in range(n)]

# Display the result

print("Arithmetic sequence:", sequence)

Output



QUESTION 3

CALCULATING VOLUME UNDER SURFACE Z = X\*\*2 + Y\*\*2 OVER THE SQUARE REGION 0<=X, Y<=1

#CODE

import numpy as np # Import NumPy for numerical operations

# Define the region bounds

a, b = 0, 1

# Choose the number of intervals (higher = more accurate)

N = 100

# Generate x and y values

x = np.linspace(a, b, N)

y = np.linspace(a, b, N)

# Create a mesh grid from x and y

X, Y = np.meshgrid(x, y)

# Compute z = x^2 + y^2

Z = X\*\*2 + Y\*\*2

# Calculate the area of each small rectangle

dx = dy = (b - a) / N

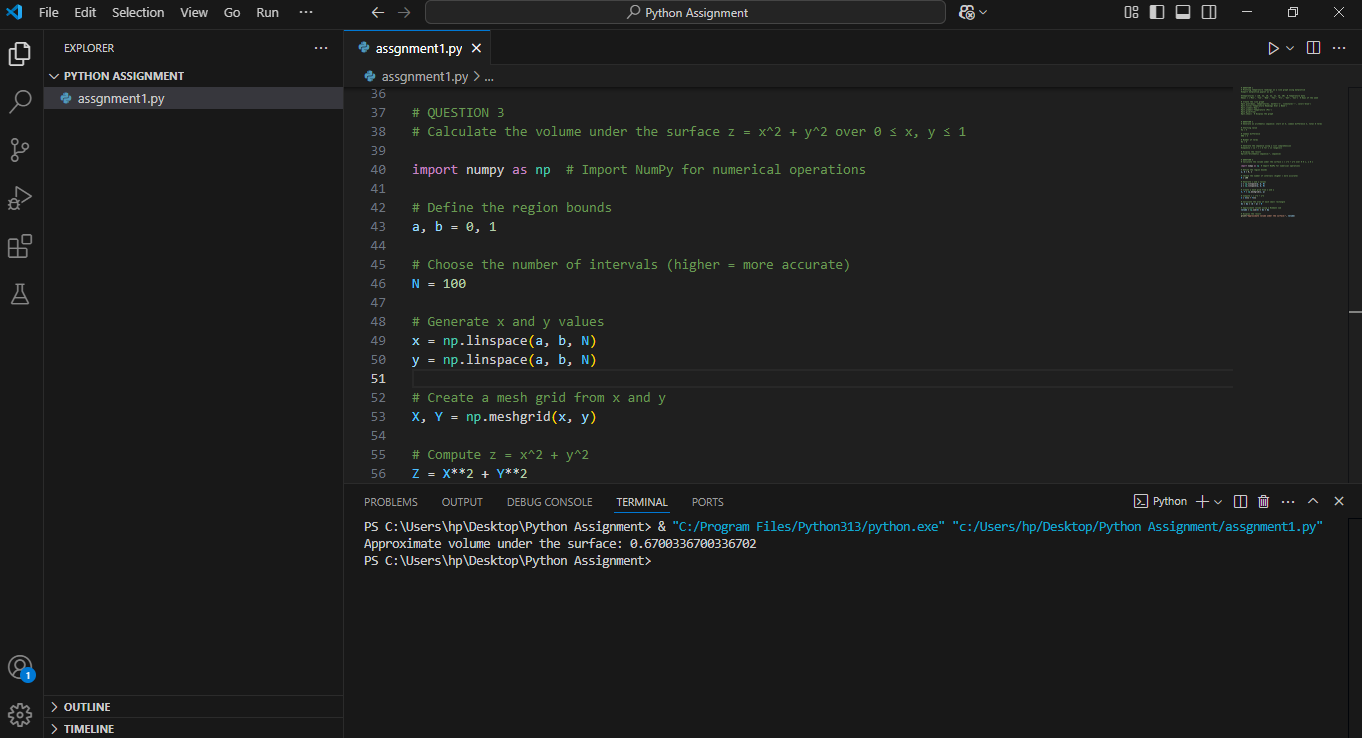
# Approximate volume using a Riemann sum

volume = np.sum(Z) \* dx \* dy

# Display the result

print("Approximate volume under the surface:", volume)

Output



QUESTION 4

Differences between Compiled and Interpreted programming languages. Classify Python in this context

Compiled Language

Source code is converted directly into machine code by a compiler before execution

Generally faster execution as the code is already translated

Errors are caught during compilation

Runs faster after compilation.

Examples are C, C++, and Rust

Interpreted Language

Source code is executed line by line by an interpreter at runtime

Generally slower than compiled languages

Easier to debug and test.

Examples are Python and JavaScript.

Errors are caught during execution

Python’s Classification

Python is an interpreted language. However, it uses a hybrid approach

Python code is first compiled to bytecode

The bytecode is then interpreted by the Python Virtual Machine (PVM)

This hybrid approach gives Python some benefits of both compiled and interpreted languages while maintaining its flexibility and ease of use.